Towards Dynamic Knowledge Support in Software Engineering Processes

Gregor Grambow[1], Roy Oberhauser[1], Manfred Reichert[2]
[1] Computer Science Dept., Aalen University, Germany
{gregor.grambow, roy.oberhauser}@htw-aalen.de
[2]Institute for Databases and Information Systems, Ulm University, Germany
manfred.reichert@uni-ulm.de

Abstract: Software development projects have historically been challenged with respect to producing a quality product. To some extent, this can be attributed to the complex, dynamic, and highly intellectual process of creating software. While efforts have been made to support both *process execution* and *knowledge management* with automated systems in software engineering (SE), the effective dissemination of knowledge and its concrete utilization in the development process remain problematic. This paper contributes an approach that associates automated workflow governance support with knowledge management and semantic technology. This enables the dynamic injection of contextually relevant SE knowledge into software development workflow execution.

1 Introduction

The process of creating software is a highly dynamic process whose support and governance is not always easy. Typically, the development of new products, concepts, or components is involved and therefore standardized automatable processes are not as suitable as in, for example, industrial production. New product development is also a knowledge-intensive task [RT99] and software processes are mostly knowledge processes [KH02]. Software engineering (SE) is still a relatively new and immature discipline, and while work has gone into integrating *knowledge* and *process management* to support SE, a comprehensive and viable solution is elusive. Currently, *process management* is typically done in a documentation–centric (e.g., Rational Unified Process) or agile fashion (e.g., Scrum) and lacks automated process support. *Knowledge management*, in turn, is crucial to enable the distribution of knowledge among different people and to keep and exploit experience gained in various projects. Supporting this with an automated system can be very beneficial [TFB98]. Important capabilities of such a system are capturing, maintaining, reusing, and transferring knowledge [TFB98]. Wikis are often used for SE knowledge management because of the easy creation and access of information [SBB08]. However, retrieval of contextually relevant information from Wikis remains difficult [SBB08]. Thus, information is captured and stored but its reuse is still problematic. This could be facilitated if knowledge use was connected with process execution.

One example application is checklists that capture information to be used at specific points in the process (e.g., source code implementation). Lacking automated support, checklist usage in SE can easily be forgotten. Even if they are used, the relevance and usability of items depends on many contextual factors like the type of component that is developed or the skill level of the engineer executing the task. To accommodate this, we provide an approach for supporting the SE process with checklists that are dynamically connected to knowledge management to provide information highly relevant to the concrete situations where they are provided. The remainder of this paper is organized as follows: Section 2 introduces the developed concept, Section 3 shows its realization, and Section 4 provides an application scenario. In Section 5, related work is briefly discussed before the paper concludes with Section 6.

2 Integration of Knowledge and Process Management

To be able to automatically establish and support the dynamic integration of knowledge management and process execution, different elements have to be in place as illustrated in Figure 1: (1) A facility to enable users to collect and store knowledge. This can be general SE knowledge or specialized information about the current development process. Further, this information should be accessible on a semantic level to permit a system to automatically utilize it. (2) Workflows that are part of the development process (e.g., a workflow for developing a new software component) should be automatically governed and supported. (3) There should be facilities to gather contextual information and to automatically process and use this information to match knowledge and process governance to various situations. Contextual information includes information about the current SE environment situation (e.g., tool use, module involved), user profile, or project. (4) The aforementioned areas must be unified or connected automatically to enable mutual access to information in the areas. E.g., contextual information can be used to provide knowledge during process execution that matches the needs of the specific user processing an activity and the current project situation.

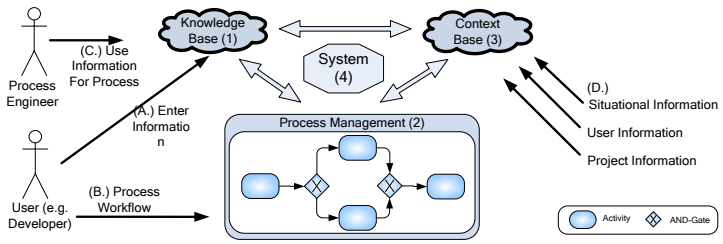


Figure 1: Process / Knowledge Management Concept

The presented concept enables the integration of knowledge directly into workflow execution, thus supporting operational influence. To limit the scope of this paper, only the use of automatically tailored checklists presented to the user at process points was selected for demonstrating the capabilities and advantages of such an automated SE knowledge integration approach.

Checklists present unique challenges because they consist of knowledge that is directly process-relevant, dynamic (e.g., items can be adjusted due to defect causal analysis), contextually dependent (some items are irrelevant in certain contexts, are activity-specific, programming-language-specific, platform-specific, etc.), and to some degree user-profile-specific (e.g., junior vs. senior engineers, database vs. GUI). To match a multitude of different situations, checklists either can be predefined and static or dynamically generated utilizing contextual information. Furthermore, checklist processing and completion can be required to complete an activity or checklist items can also be optionally used as additional information (guidance). Superimposing a checklist scenario on Figure 1, (A) during the execution of projects, users can add information to the knowledge base, e.g., when encountering problems and finding solutions for them. They can tag this information to support later discovery by humans or any automated system. Examples of tags on that information include 'junior' to indicate applicability for junior engineers or 'backend' to relate them to a specific implementation area. (B) Project execution is managed and governed automatically by the system by means of different workflows belonging to the development process. Examples of activities governed that way include 'Implement Solution' where new source code is developed or 'Run Developer Test' where source code is tested by the developer. (C) These workflows can be annotated, e.g., by a process engineer at specific points to use checklists (e.g., requirements or testing checklists). The checklists can be easily predefined in the knowledge base by tagging information with checklist tags. (D) The system continuously detects new facts about the current situation and stores them in the context base. This is enabled by a set of sensors integrated in various SE tools that automatically provide information about tool and artifact usage. An example for such a detected event can be the modification of a source code artifact in an IDE. Utilizing this situational information, dynamic checklists are supported - workflows can be annotated to include checklists at certain points, but these do not have to be predefined. Such a dynamic checklist is automatically generated by the system matching information of the current situation as, e.g., the skill level of the user or the time and quality constraints of the project using tags on information in the knowledge base.

3 Concept Realization

The presented concept has been realized within the CoSEEEK [GOR11] framework that comprises different modules to enable automated SE project support. The described knowledge base is realized by a semantic mediawiki [KVV06] that is queried using SPARQL [PS06]. This is enabled via the 'SparqlExtension' plugin and the SPARQL processor Joseki [Mc02]. Dynamic process governance in CoSEEEK's process management module is enabled by AristaFlow [DR09], a BPM suite supporting adaptations of running workflow instances. The context base is realized within CoSEEEK's context management module by an OWL-DL [WW04] ontology that is processed by the reasoner Pellet [Si06] and accessed programmatically via the Jena framework [Mc02]. Event management is realized utilizing the Hackystat framework [Jo07] that provides sensors for event detection and Esper [Es11] for event processing to derive higher-level events from detected low-level events.

To enable the system to utilize context information directly for process execution and to unite the latter with knowledge management, the process management concepts are semantically annotated within the ontology as illustrated in Figure 2.

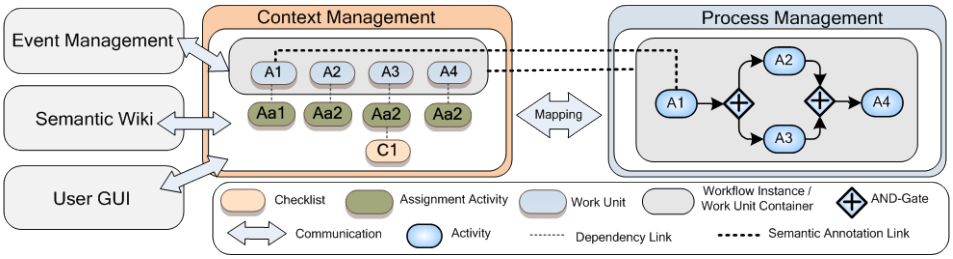


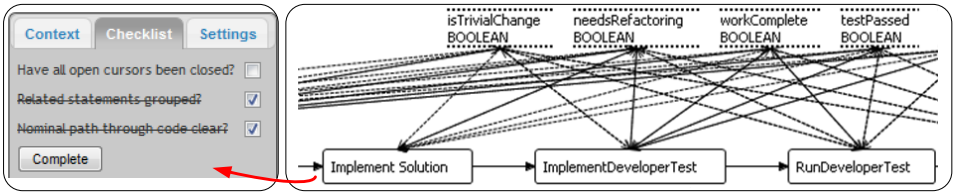
Figure 2: Process / Knowledge Management Realization

A so-called *WorkUnitContainer* exists for each workflow in the ontology, and for each activity, there is a *WorkUnit*. Utilizing these concepts, the context management module encapsulates the process management module and connects workflow execution with the other modules. Furthermore, the workflow modeling is extended in the context management module by so-called *AssignmentActivities*. These explicitly represent human tasks (e.g., 'Implement Developer Test') and can be connected via *Checklist* concepts. These, in turn, indicate that the respective activity has an assigned checklist. Checklists are of two types: static and dynamic. A static checklist has a type that is matched by the system to a predefined checklist in the semantic mediawiki. A dynamic checklist can have various tags instead of a type that are used to query items from the wiki that match these tags via SPARQL. When a *WorkUnit* becomes active, the assigned *AssignmentActivity* is provided to the user via a special GUI. If that *AssignmentActivity* has a *Checklist,* the latter is also presented to the user on a separate tab in the GUI as depicted in Figure 3a. It is also configurable via the *Checklist* concept in the ontology whether all checklist items must be explicitly checked by the user to complete the activity or if it only provides optional information. Semantic web technology is utilized here as extension to process management not only for the automated provision of checklists but for comprehensive process support including features like automated integration of software quality measures into the operational process (see [GOR11]).

4 Scenario Application

A scenario illustrating the application of the presented concept to an SE project is now described. The project utilizes the OpenUP SE process that provides different workflows for SE structured in an iterative approach. Figure 3b shows a snippet of the 'Develop Solution Increment' workflow modeled in AristaFlow that deals with the development of software with contained activities like 'Implement Solution'. In this scenario, a developer notices that in the database code of the developed application opened cursors were not always properly closed, resulting in erroneously locked resources. After fixing the problem, she puts a note into the wiki that indicates that developers should be cautious with open cursors and tags it with 'Backend' and 'Development'.

Assume that a process engineer has added a dynamic checklist to the 'Implement Solution' activity and tagged it with 'Development'.



(a) Dynamic Checklist (b) Workflow Snippet

Figure 3: Process / Knowledge Management Application Example

That way the information can be provided automatically to the developers. This is illustrated in Figure 3. Furthermore, the system can detect by its sensors to which component the processed artifacts belong and thus only provide that checklist item for backend development. To automatically adapt checklists to show the most important items, a rating mechanism will be integrated enabling the users to rate the items.

5 Related Work

In related work, few approaches directly connect SE knowledge with automated process execution. In [Li03], a study is presented reviewing various approaches to knowledge management systems. These range from knowledge-based systems over data mining systems to expert systems providing decision support. One example is the work presented in [Ns+02] that proposes an extension to data warehouses called knowledge warehouse to facilitate capturing as well as retrieving and sharing knowledge. These approaches solely focus on the improvement of knowledge management technology. In contrast, CoSEEEK features an active connection of captured knowledge with context information and with the executed process to provide active assistance utilizing that knowledge. There are other approaches utilizing knowledge for process execution. In [WR+09], a case base and case-based reasoning techniques are used for reusing knowledge on how to deal with exceptional situations during process execution. [MTB07] follows a similar approach: A case base is used to facilitate retrieval of past workflows to provide users with authoring support for the creation of new variants.

6 Conclusion

In the dynamic SE domain it is still challenging to provide knowledge-based support for the operational process. In this area, this work contributes an approach for connecting and automating *knowledge* and *process management*. Semantic technology is used as link between automatically gathered context information, knowledge resources, and process execution. Thus, it becomes possible to dynamically assemble knowledge relevant to the executing user and to automatically and seamlessly integrate this knowledge with the users' current workflow.

Future work includes scaling the retrieval and checklist composition to include various non-local repositories, envisioning general SW engineering checklist repositories that utilize social feedback mechanisms to filter and prioritize items.

Acknowledgement

This work was sponsored by BMBF (Federal Ministry of Education and Research) of the Federal Republic of Germany under Contract No. 17N4809.

References

[DR09] Dadam, P., Reichert, M.: The ADEPT Project: A Decade of Research and Development for Robust and Flexible Process Support - Challenges and Achievements. Springer, Computer Science - Research and Development, 23(2): 81-97, 2009

[Es11] http://www.espertech.com/products/esper.php [retrieved Apr. 2011].

[Jo07] Johnson, P.M.: Requirement and Design Trade-offs in Hackystat: An In-Process Software Engineering Measurement and Analysis System. Proc. 1st Int. Symp. Empirical SW Engineering and Measurement, IEEE Comp. Society, pp. 81-90, 2007

[Li03] Liao, S.: Knowledge management technologies and applications--literature review from 1995 to 2002. Expert systems with applications, 25: 155-164, 2003.

[KH02] Kess, P., Haapasalo, H.: Knowledge creation through a project review process in software production. Int'l Journal of Production Economics, 80(1). 49-55, 2002

[KVV06] Krötzsch, M., et al.: Semantic mediawiki. The Semantic Web-ISWC 2006, 2006

[Mc02] McBride, B.: Jena: A Semantic Web Toolkit. Internet Computing, 6(6): 55-59, 2002.

[MTB07] Minor, M., Tartakovski, A., Bergmann, R.: Representation and Structure-Based Similarity Assessment for Agile Workflows. LNCS, 4626: 224-238, 2007

[Ns+02] Nemati, H. R., Steiger, D. M., Iyer, L. S., & Herschel, R. T.: Knowledge warehouse: an architectural integration of knowledge management, decision support, artificial intelligence and data warehousing. Decision Support Systems, 33: 143–161, 2002

[GOR11] Grambow, G., Oberhauser, R., Reichert, M.: Contextual Injection of Quality Measures into Software Engineering Processes. Int'l Journal on Advances in Software (to appear)

[PS06] Prud'hommeaux, E., and Seaborne, A.: SPARQL Query Language for RDF. W3C WD 4 October 2006.

[Si06] Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL Reasoner. Journal of Web Semantics, 5(2): 51-53, 2006.

[RT99] Ramesh, B., Tiwana, A.: Supporting collaborative process knowledge management in new product development teams. Decision support systems 27: 213-235, 1999

[SBB08] Schaffert, S., et al.: Semantic wikis. IEEE software, 25(4): 8-11, 2008

[TFB98] Teigland, R.E., Fey, C., Birkinshaw, C.: Knowledge Dissemination. Global R&D Operations: Case Studies in Three Multinationals in the High Technology Electronics Industry, Stockholm School of Economics, Stolkholm, 1998.

[WR+09] Weber, B., Reichert, M., Wild, W., Rinderle-Ma, S.: Providing Integrated Life Cycle Support in Process-Aware Information Systems. Int'l Journal of Coop. Inf. Systems, 18(1): 115-165, 2009.

[WW04] World Wide Web Consortium: OWL Web Ontology Language Semantics and Abstract Syntax, (2004) [retrieved Apr. 2011]